NAVAL POSTGRADUATE SCHOOL
Monterey, California

DTIC
S ELECTE
JUL 2 7 1993
B D

THESIS

AN ANALYSIS OF EFFECTS OF VARIABLE
FACTORS ON WEAPON PERFORMANCE
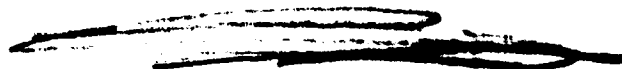
by

William V. Ridings

March 1993

Thesis Advisor: Harold J. Larson

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden. to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE March 1993	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE

An Analysis of Effects of Variable Factors in Weapon Performance

5. FUNDING NUMBERS

6. AUTHOR(S)

RIDINGS, William V.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release, distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Statistical analysis provides a powerful tool for modern decision makers. Unfortunately, this tool can be a two-edged sword. Improper or erroneous analysis can result in incorrect and costly decisions. Many analysis errors can be traced to the misapplication of statistical methods.

When examining experimental data, it is first necessary to determine the true nature of that data, specifically, the structure from which the data is drawn. This determination will then be a primary factor in the choice of statistical tests.

This thesis examines an analysis performed by Surface Warfare Development Group (SWDG). The SWDG analysis is shown to be incorrect due to the misapplication of testing methods. A corrected analysis is presented and recommendations suggested for changes to the testing procedures used by SWDG. Additionally, a computer program to perform basic Analysis of Variance (ANOVA) tests is provided to be appended to the current SWDG statistical software.

14. SUBJECT TERMS ANOVA, Categorical, Pearson, Statistics	15. NUMBER OF PAGES 61
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

AN ANALYSIS OF EFFECTS OF VARIABLE
FACTORS ON WEAPON PERFORMANCE

by

William V. Ridings
Lieutenant, United States Navy
B.S. Mathematics, U.S. Naval Academy, 1985

Submitted in partial fulfillment
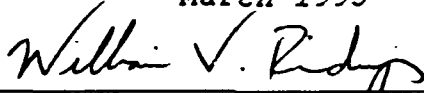of the requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
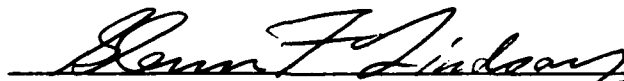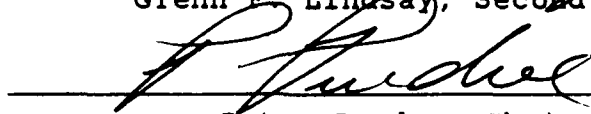March 1993

Author: _____
William V. Ridings

Approved by: _____
Harold J. Larson, Thesis Advisor

_____
Glenn F. Lindsay, Second Reader

_____
Peter Purdue, Chairman
Department of Operations Research

ii

ABSTRACT

Statistical analysis provides a powerful tool for modern decision makers. Unfortunately, this tool can be a two-edged sword. Improper or erroneous analysis can result in incorrect and costly decisions. Many analysis errors can be traced to the misapplication of statistical methods.

When examining experimental data, it is first necessary to determine the true nature of that data, specifically, the structure from which the data is drawn. This determination will then be a primary factor in the choice of statistical tests.

This thesis examines an analysis performed by Surface Warfare Development Group (SWDG). The SWDG analysis is shown to be incorrect due to the misapplication of testing methods. A corrected analysis is presented and recommendations suggested for changes to the testing procedures used by SWDG. Additionally, a computer program to perform basic Analysis of Variance (ANOVA) tests is provided to be appended to the current SWDG statistical software.

DTIC QUALITY INSPECTED 3

Accession For	
NTIS GRA&I	☑
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

iii

THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

TABLE OF CONTENTS

I. BACKGROUND

A. ORGANIZATION

Surface Warfare Development Group (SWDG) is responsible for the Ship ASW Readiness/Effectiveness Measuring Program (SHAREM), a Chief of Naval Operations sponsored effort established in 1969 to quantitatively assess the antisubmarine warfare (ASW) performance of surface ships. This program was expanded in 1973 to include surface ASW tactics. SWDG conducts several large-scale ASW exercises called SHAREMs each year in order to gather data on ship, submarine, and weapons system performance. These exercises are conducted in various regions throughout the world, typically involve seven to ten combatant platforms and supporting aircraft, and extend for seven to fourteen days. Data is gathered on all involved platforms to be later collated and processed by SWDG. The goals of the SHAREM program are met through the design, conduct, reconstruction, and analysis of at-sea exercise data. [Ref. 1:p. 1]

B. DATA ANALYSIS

SWDG uses a statistical package created on-site to perform analyses on SHAREM exercise data. This package is designed to be used by personnel who are not generally familiar with advanced statistical tests. The SWDG package is not complete,

1

however. Tests such as Analysis of Variance (ANOVA) are being performed by hand by senior analysts. SWDG wishes to add an ANOVA module to its statistical package.

C. WEAPON SYSTEM

The weapon system examined herein is a real-world system that will be referred to as XYZ, to keep this treatment unclassified at the request of SWDG. SWDG, in conjunction with Operational Test and Evaluation Force (OPTEVFOR), performed an analysis of the factors affecting the performance of this system in early 1992. The original SWDG analysis is classified Top Secret and was released only to those agencies considering XYZ for procurement. In fact, the SWDG analysis was a contributing factor in the decision to proceed with XYZ procurement.

D. PROBLEM OVERVIEW

System XYZ was tested over a period of nine months, using different platforms and several geographical areas. From the raw data, SWDG eliminated those trials in which there was a mechanical failure of the weapon, interference from platforms not associated with the trial, breakdown of the data recording equipment, or incomplete data compilation. From the original 190 trials, 89 were considered usable for analysis purposes. These trials are assumed to be independent. In fact, they may

not be; however, the data as tabulated before analysis gives no indication of being correlated. Data resolution was such that four factors that might have an effect on XYZ performance could be examined. These factors will hereafter be referred to as A, B, C, and D for classification purposes as previously discussed. Additionally, each of these factors could inherently be broken down into two levels, high and low. Finally, the test result (decision variable) was documented as weapon success or weapon failure. A weapon was considered su⁻cessful only if it impacted the target; otherwise, it was considered a failure. A breakdown of the analysis data is found in Table 1.

TABLE 1: COMPILATION OF ANALYSIS DATA

SUCCESSES FAILURES		HI A		LOW A	
		HI B	LOW B	HI B	LOW B
HI	HI D	0/0	0/6	0/3	5/4
C	LOW D	0/0	1/4	7/3	2/1
LOW	HI D	0/0	4/9	0/14	2/5
C	LOW D	0/0	3/5	5/3	1/2

Several details are of immediate note. First, there is no data available for the factor combination of high A/high B. No explanation is given for the absence of this data, although it is conceivable that SWDG or OPTEVFOR considered this

3

combination as tactically infeasible. Secondly, there is an unequal number of replications of each factor combination for which there is data. Two combinations have only three associated trials, while one has fourteen trials. Finally, there are factor combinations with no successes, only failures. These details must be taken into account in order to properly perform analysis.

E. PURPOSE

This thesis proposes that the analysis performed by SWDG on the factors affecting system XYZ is flawed and offers a corrected analysis. Additionally, an interim ANOVA program is included that can be added to the current SWDG package.

II. SWDG ANALYSIS

A. ANALYSIS APPROACH

The analysts at SWDG viewed this problem as a 2^4 factorial Analysis of Variance (ANOVA) problem. There are 16 possible combinations of factor levels in this approach. To determine the value associated with each combination, a weapon success was given a value of ten and a weapon failure a zero. Each cell value was then calculated by

$$Cell\ Value = \frac{10 \times number\ of\ successes\ within\ cell}{number\ of\ weapons\ fired\ within\ cell}. \qquad (2.1)$$

Those cells without data were given a value of zero. Given these values, ANOVA calculations were made with the probability of Type I error equal to 0.05. It was assumed that the higher order interactions (third and fourth order) were not significant. These values were pooled with sampling error to estimate the residual effect. The results are displayed in Table 2.

As shown, the SWDG analysis found that main effect D and the BD interaction are significant at the 0.05 level, which is the SWDG standard for Type I error.[Ref. 1:p. 3]

5

TABLE 2: ANALYSIS OF VARIANCE

SOURCE	DF	SS	MS	F
A	1	21.71	21.71	1.07
B	1	<1	<1	0.05
C	1	27.16	27.16	1.34
D	1	127.44	127.44	6.29[t]
AB	1	9.45	9.45	0.47
AC	1	29.70	29.70	1.47
AD	1	44.10	44.10	2.18
BC	1	38.69	38.69	1.91
BD	1	110.79	110.79	5.47[t]
CD	1	<1	<1	0.05
3X/4X & ERROR	78	1579.72	20.25	
TOTAL	88	1988.76		

Significant with α =0.05

B. ANOVA REVIEW

ANOVA is used to test hypotheses about the equality of means of samples from three or more normal populations. [Ref. 2:p. 492] The ANOVA procedure determines whether the discrepancies between the sample means are greater than could reasonably be expected from the variation that occurs within the sample classification.[Ref. 3:p 167] It is important to recognize that the ANOVA procedure requires independent samples from normal populations in order to produce correct results.

C. CRITIQUE OF SWDG ANALYSIS

In reviewing the SWDG analysis, it was noted that this problem was not a "standard" 2^4 factorial ANOVA design as SWDG claimed. It is, in fact, a 3/4 replication of a 2^4 factorial arrangement with unequal sample sizes. In order to make the observed data fit the 2^4 factorial form, SWDG used zeroes for the cells with no observations. There is no acceptable rationale for this procedure.

Additionally, successes were valued as tens and failures as zeroes. These values cannot be viewed as random variables from a normal population. They are more accurately described as the results of a binomial experiment consisting of n = 89 independent Bernoulli trials with probability of success p. The value of p can be estimated by

$$\hat{p} = \frac{number\ of\ successes}{number\ of\ trials} \qquad (2.2)$$

Application of the central limit theorem reveals that given a "large" number of trials, the distribution of the results of this experiment is approximately normal.
[Ref. 2:p. 294] The question remains as to whether 89 trials is sufficiently large to justify the application of the ANOVA procedure. For these reasons, other statistical methods should be explored.

III. ALTERNATIVE ANALYSIS

A. CATEGORICAL DATA ANALYSIS

Statistical methodology for categorical data analysis traces its roots to the work of Francis Galton in the 1880's on regression methods for continuous variables. The early literature on categorical data analysis dealt primarily with how to measure association. M. H. Doolittle's paper in 1887 on the subject contained the following quote:

> Having given the number of instances respectively in which things are both thus and so, in which they are thus but not so, in which they are so but not thus, and in which they are neither thus nor so, it is required to eliminate the general quantitative relativity inhering in the mere thingness of the things, and to determine the special quantitative relativity subsisting between the thusness and the soness of the things.[Ref. 4:p. 28]

Karl Pearson and G. Udny Yule made significant contributions to the study of association between categorical variables at the turn of the twentieth century. Although they differed in opinion regarding continuous distributions underlying the data structures, their work laid the groundwork for modern categorical analysis.[Ref. 4:p. 26-7]

B. PEARSON CHI-SQUARED TEST

In 1900, Pearson was examining various gambling games in Monte Carlo. Through the process of analyzing whether

possible outcomes on a roulette wheel were equally likely, he proposed the test statistic

$$X^2 = \sum \frac{(n_i - m_i)^2}{m_i} \ . \qquad (3.1)$$

where m_i is the expected cell frequency for cell i, n_i is the observed cell count, and N is the total number of cells. [Ref. 4:p. 43] For large samples, this statistic approximates a chi-squared distribution with N - 1 degrees of freedom.

C. CONTINGENCY TABLES

Table 3 is a representation of factor A from the XYZ system data in the format commensurate with categorical data analysis.

TABLE 3:   FACTOR A BREAKDOWN

EFFECT LEVEL	SUCCESSES	FAILURES	TOTAL
HI A	8	24	32
LOW A	22	35	57
TOTAL	30	59	89

Tables of this form, with each cell containing the frequency count of the outcome, are called contingency tables, a term introduced by Pearson in 1904.[Ref. 4:p. 9] This is an

9

example of a two-way contingency table, because there are two categorical response variables with number of levels I = 2 and J = 2, respectively.

D. ANALYSIS OF SYSTEM XYZ

1. Modification of Pearson statistic

Statistical independence of the categories can be examined through a modification of Pearson's original test. First, expected cell frequencies are estimated by

$$\hat{m}_{ij} = \frac{Total_{i.} \times Total_{.j}}{n} . \qquad (3.2)$$

The Pearson chi-squared statistic can then be calculated as

$$X^2 = \sum \sum \frac{(n_{ij} - \hat{m}_{ij})^2}{\hat{m}_{ij}} . \qquad (3.3)$$

Pearson stated that X^2 approximated a chi-squared distribution with IJ − 1 degrees of freedom.[Ref. 4:p. 43] This was disputed in 1922 by R. A. Fisher, who proved that the correct degrees of freedom is

$$df = (IJ-1) - (I-1) - (J-1) = (I-1)(J-1) . \qquad (3.4)$$

Fisher's result will be used in this analysis.[Ref. 5:p. 213]

10

TABLE 4: CONTINGENCY TABLE WITH PEARSON STATISTICS

MAIN EFFECT	SUCCESS	FAILURE	TOTAL
HIGH A	0.720 10.787 8	0.366 21.213 24	32
LOW A	0.404 19.213 22	0.205 37.787 35	57
TOTAL	30	59	89

2. Presentation of values

The standard format for the presentation of these calculations is shown in Table 4. The estimated cell value m_{ij} is inserted in the top right corner of the cell and the individual cell contribution to the Pearson statistic is placed in the top left corner.

3. Main Effects

The Pearson statistic can be used to test the hypothesis that there is no relationship between the variables. In the examination of the main effects of XYZ, this translates into a test of the relationship between the different effect levels. As the data is presented in a two-way contingency table with two levels per variable, the corresponding Pearson chi-square statistic has

$$(2 - 1)(2 - 1) = 1 \textit{ degree of freedom.} \qquad (3.5)$$

11

Using the SWDG standard of 0.05 for TYPE I error, the appropriate chi-square quantile is

$$\chi^2_{1,0.95} = 3.841. \qquad (3.6)$$

Therefore, if the Pearson statistic calculated for each of the main effects is greater than 3.841, there is a significant difference in the variables due to the level of the effect. For example, using the values from Table 4, the Pearson statistic for main effect A is

$$X^2 = 0.720 + 0.404 + 0.366 + 0.205 = 1.695 . \qquad (3.7)$$

Since 1.695 is less than 3.841, the hypothesis of independence cannot be rejected. Main effect A, therefore, is not significant at the 0.05 level. Table 5 is a summary of the results of the examination of the main effects. The contingency tables for all main effects are found in Appendix A.

TABLE 5:   MAIN EFFECTS SUMMARY

EFFECT	χ^2	SIGNIFICANCE
A	1.695	Not significant
B	0.008	Not significant
C	1.714	Not significant
D	8.822	Significant

Probability of Type I error = 0.05

In the examination of the main effects, it appears that Pearson's chi-square test and ANOVA yield the same result. Only main effect D is significant at the 0.05 level. Table 6 provides a comparison of the results. Although there is also a similar ordering of the effects with regard to the magnitude of the test statistics, this does not provide justification for the use of the ANOVA procedure. This will be borne out in the analysis of the interactive effects.

TABLE 6: ANOVA-PEARSON COMPARISON

EFFECT	ANOVA		PEARSON	
	F	SIGNIFICANCE	x^2	SIGNIFICANCE
B	0.05	Not sig.	0.008	Not sig.
A	1.07	Not sig.	1.695	Not sig.
C	1.34	Not sig.	1.714	Not sig.
D	6.29	Signif.	8.822	Signif.

Probability of Type I error = 0.05

4. Interactive Effects

Calculation of interactive effects using Pearson's statistic are somewhat less straightforward. The data is first broken down to reflect effect combinations, as shown in Table 7.

If the Pearson statistic was calculated at this point, the resulting value would not measure the interactive effect. It would test the independence of the four combinations versus success and failure, and would have 3 degrees of freedom.

13

These three degrees of freedom include the two main effects and the interactive effect. The interactive effect must be isolated.

TABLE 7: AC INTERACTION FIRST STAGE

EFFECT	SUCCESSES	FAILURES	TOTAL
HI A/HI C	1	10	11
HI A/LOW C	7	14	21
LOW A/HI C	14	11	25
LOW A/LOW C	8	24	32
TOTAL	30	59	89

In order to find the interactive effect, it is useful to examine the different levels as contrasting coefficients. Let '+' denote a high level effect, and '-' a low level effect. This translates into a coefficient table, such as Table 8.

TABLE 8: TABLE OF COEFFICIENTS

EFFECT	1st COEFF MAIN EFFECT A	2nd COEFF MAIN EFFECT C	PRODUCT AC INTERACTION
HI A/HI C	+	+	+
HI A/LOW C	+	-	-
LOW A/HI C	-	+	-
LOW A/LOW C	-	-	+

14

If the rows with a "positive" first factor effect, in this case one and two, were combined, and if the "negative" first factor rows were combined, the resulting table is the same as the contingency table for main effect A. Similarly, if the positive second factor rows were merged, as well as the negative second factor rows, the resulting table would represent main effect C.

The product column of the table of coefficients is the result of multiplying the two factor columns. It would be reasonable, therefore, to combine the positive and negative rows of the product column to determine the interaction between the two factors. Table 9 reflects this merger.

TABLE 9: AC INTERACTION SECOND STAGE

EFFECT	SUCCESSES	FAILURES	TOTAL
HI A/HI C LOW A/LOW C	9	34	43
HI A/LOW C LOW A/HI A	21	25	46
TOTAL	30	59	89

The Pearson chi-squared test is now applied. Again, the resulting test statistics are compared to a chi-square distribution with one degree of freedom at the 0.05 level. A summary of the results is found in Table 10. The contingency tables for all two-way interactive effects are found in Appendix B.

TABLE 10: INTERACTIVE EFFECTS SUMMARY

EFFECT	x^2	SIGNIFICANCE
AC	6.079	Signif.
AD	4.088	Signif.
BC	0.946	Not sig.
BD	5.504	Signif.
CD	0.483	Not sig.

Probability of Type I error = 0.05

The AB interaction deserves special attention, because the high A/high B effect combination has no associated data. Consequently, there is no way to make a good approximation of the AB interaction. The methodology used for the other two-way interactions may not yield correct solutions when applied in this case, due to the loss of a degree of freedom. The three effect combinations are examined in Table 11, giving

$$\chi^2 = 2.45 \; . \qquad (2 \; df) \qquad\qquad (3.8)$$

The two degrees of freedom for this value represent a mixture of the single degrees of freedom for A, B, and AB. When 2.45 is compared with the sum of the chi-square values for A and B,

$$1.695 + 0.008 = 1.703 \; , \qquad\qquad (3.9)$$

it appears that the AB interaction is not a significant effect.

TABLE 11: AB COMBINATIONS

EFFECTS	SUCCESSES		FAILURES		TOTAL
HI A/LOW B	0.720	10.787	0.366	21.213	32
		8		24	
LOW A/HI B	0.003	11.798	0.002	23.202	35
		12		23	
LOW A/LOW B	0.901	7.416	0.458	14.584	22
		10		12	
TOTAL	30		59		89

$$X^2 = 0.720 + 0.003 + 0.901 + 0.366 + 0.002 + 0.458 = 2.45$$

As with the main effects, the results of the Pearson test on interactions can be compared to the SWDG ANOVA results. This comparison is made in Table 12.

TABLE 12: ANOVA-PEARSON COMPARISON (INTERACTIONS)

EFFECT	ANOVA		PEARSON	
	F	SIGNIFICANCE	x^2	SIGNIFICANCE
AB	0.47	Not sig.	----	-------
AC	1.47	Not sig.	6.079	*Significant*
AD	2.18	Not sig.	4.088	*Significant*
BC	1.91	Not sig.	0.946	Not sig.
BD	5.47	Significant	5.504	Significant
CD	0.05	Not sig.	0.483	Not sig.

Probability of Type I error = 0.05

The Pearson chi-squared test reveals that three interactive effects, AC, AD, and BD, are significant at the 0.05 level. ANOVA shows only BD as being significant.

IV. CONCLUSIONS AND RECOMMENDATIONS

A. COMPARISON OF TESTS

The ANOVA procedure is not the best choice of methods to apply to the XYZ system data. The data is somewhat unwieldy; the unequal number of trials per effect combination, cells with a very small number of trials, and cells with no trials are significant problems. The use of zeroes in the latter case to fill out the 2^4 factorial design is a major error. Furthermore, the ANOVA procedure is premised upon continuous response variables, whereas the data in this case is discrete.

SWDG's calculations employ formulas which assume equal sample sizes. They are not correct for the unequal cell-sized problem presented herein and are another factor to be considered when accounting for the discrepancies in the results of the two tests.

Contingency tables may be a better analytical tool to apply to count data such as for system XYZ. The Pearson statistic associated with contingency tables reveals two significant interactions that are missed by ANOVA, the AC and AD interactions. The small sample size and the improper formula application previously addressed are the primary cause of these omissions. Table 13 is a compilation of the results of both tests.

It is not overly surprising that the use of contingency tables was overlooked in the analysis of this problem.

TABLE 13: COMPILATION OF RESULTS

EFFECT	SIGNIFICANCE	
	ANOVA	**PEARSON TEST CONT. TABLES**
A	Not significant	Not significant
B	Not significant	Not significant
C	Not significant	Not significant
D	SIGNIFICANT	SIGNIFICANT
AB	Not significant	------
AC	Not significant	SIGNIFICANT
AD	Not significant	SIGNIFICANT
BC	Not significant	Not significant
BD	SIGNIFICANT	SIGNIFICANT
CD	Not significant	Not significant

Probability of Type I error = 0.05

Outside of Pearson's work, there was very little work done in the refinement of this form of statistical methodology until the last thirty years.[Ref. 4:p. 1] ANOVA received the majority of the attention through the first half of the of the twentieth century, and it has become the method of choice for many analysts. Although it is a powerful tool, it must be used in the proper context.

B. DESIGN OF EXPERIMENT

In retrospect, it is easy to say that the design of this experiment has a few problems, if in fact there was a design. There did not appear to be an effort taken to balance the

number of trials conducted for each effect combination. For completeness, trials should have been purposely done for all sixteen combinations. It is recognized that it is difficult in an open ocean environment to coordinate air, surface, and subsurface units in such a way as to adequately reproduce specific trials, and there may be physical limitations to possible effect combinations.

The first and perhaps most important criteria to consider when designing an experiment is the response variable. To a great extent, weapon systems provide a discrete response: hit or miss, success or failure. A continuous response variable associated with weapon systems is miss distance. Miss distance may be difficult to measure for many systems, but it can often be estimated. If the miss distances had been recorded for system XYZ, the resulting distribution could possibly have been normalized, setting the stage for the use of ANOVA.

More importantly, if miss distances can be recorded, fractional factorial experiments can be employed. A fractional factorial experimental design would necessitate a considerably smaller number of required trials, thereby generating a considerable savings in time and money. For example, the full 2^4 factorial design requires (16 * n) runs, where n is the number of desired replications. A 2^{4-1} fractional factorial design, or half-fraction design, would require only (8 * n) trials.[Ref. 3:p. 378]   Table 14

describes the trials required to meet the conditions of a half-fraction with resolution IV.

TABLE 14: HALF-FRACTION DESIGN

LEVEL AND EFFECT COMBINATIONS
1. Low A, Low B, Low C, Low D
2. Hi A, Low B, Low C, Hi D
3. Low A, Hi B, Low C, Hi D
4. Hi A, Hi B, Low C, Low D
5. Low A, Low B, Hi C, Hi D
6. Hi A, Low B, Hi C, Low D
7. Low A, Hi B, Hi C, Low D
8. Hi A, Hi B, Hi C, Hi D

It must be noted that fractional designs appear to give the user "something for nothing"; that is, no loss of information with fewer number of trials. This is not entirely true. The price paid for using the fractional design is the confounding or confusing of effects.[Ref. 3:p. 381] However, with the proper designs and the previously stated assumption of insignificant higher-order interactions, this penalty is negligible. With shrinking budgets and weapons increasing in complexity and cost, it is imperative that testing procedures be as efficient as possible to minimize expenditures.

V. ANOVA PROGRAM

A. CONSTRUCT

The ANOVA program found in Appendix C was created to be appended to the existing SWDG computerized statistical package. It is written in Borland C++ and is constructed to be user friendly, although a basic understanding of the precepts of ANOVA testing is required. This program allows the user to choose the desired test and significance level, prompts the user for data input, performs the necessary ANOVA calculations, and presents the results in tabular form. Table look-ups are eliminated as this program calculates the appropriate F statistic quantiles as required for significance testing. The ANOVA tests provided are restricted in scope, as this program is intended as only an interim analytical aid until a sophisticated statistical package meeting SWDG's needs can be procured.

B. CAPABILITIES

This program performs ANOVA calculations for the following tests: One-way ANOVA, Two-way ANOVA, and 2^k factorial ANOVA with k <= 5.

C. METHODOLOGY

1. Models

ANOVA is used to test hypotheses about the equality of means of two or more normal populations, using sample data drawn from these populations. The simplest models assume that each data observation can be expressed as a sum of a mean value, a value or values attributed to effects, and a term for sampling error.   These models are referred to as *completely randomized* models.   The ANOVA procedures used herein are based on these models.

a.   One-way ANOVA

The model for one-way ANOVA is

$$Y_{ij} = \mu + \tau_i + \epsilon_{ij} \tag{5.1}$$

where Y_{ij} are the observed values for the experiment, mu is the overall mean of the data, theta is the average deviation for each i, and e_{ij} are normally distributed and independent errors of observation.[Ref 2:p. 495] It is not necessary for there to be an equal number of observations from each population. The test will determine if there is a significant difference between the means of the populations.

b.   Two-way ANOVA

The model for two-way ANOVA is

$$Y_{ij} = \mu + \tau_i + \beta_j + \epsilon_{ij}. \tag{5.2}$$

24

This *randomized block* model uses the same variables as the one-way model, with the addition of a variable B_j to represent the expected deviation caused by an extraneous condition. [Ref. 2:p. 499] For this design, an equal number of observations are required from each population. Additionally, this model assumes no interaction effect between the populations and the extraneous condition. The removal of the variability of the outside factor gives a more sensitive test for the equality of the population means, while at the same time testing that the outside factor is equally applied to each population.[Ref. 2:p. 503-4]

c. 2^k *Factorial ANOVA*

2^k factorial designs examine the effects of multiple factors, each at two levels, on a continuous, normally distributed response variable. Yates' algorithm is utilized as the computational method. This algorithm provides a rapid calculation of effects using a standard ordering of the data.[Ref. 3:p. 342] Yates' algorithm estimates significance of main effects and interactions. Equal numbers of replications are required for this program, and there must be data for every factor combination. This application is limited to the examination of k = 5 factors and 20 replications per cell. The data must be placed into standard order when input to the program.

2. F Statistic

The F statistic used by this program is derived from an APL program created by Professor H. J. Larson at the Naval Postgraduate School, which is based on a method created by A. H. Carter in 1947[Ref. 6:p. 352-7]. The program eliminates the need for table look-ups by calculating the appropriate F distribution quantile.

APPENDIX A: MAIN EFFECTS

TABLE A-1: MAIN EFFECT A

EFFECT	SUCCESSES		FAILURES		TOTAL
HI A	.720	10.787	.366	21.213	32
		8		24	
LOW A	.404	19.213	.205	23.202	57
		22		35	
TOTAL	30		59		89

$$Q = .720 + .404 + .366 + .205 = 1.695$$

TABLE A-2: MAIN EFFECT B

EFFECT	SUCCESSES		FAILURES		TOTAL
HI B	.003	11.798	.002	23.202	35
		12		23	
LOW B	.002	18.202	.001	35.798	54
		18		36	
TOTAL	30		59		89

$$Q = .003 + .002 + .002 + .001 = .008$$

TABLE A-3: MAIN EFFECT C

EFFECT	SUCCESSES		FAILURES		TOTAL
HI C	.676	12.1	.344	23.865	36
		15		21	
LOW C	.460	17.865	.234	35.135	53
		15		38	
TOTAL	30		59		89

$$Q = .676 + .460 + .344 + .234 = 1.714$$

TABLE A-4: MAIN EFFECT D

EFFECT	SUCCESSES		FAILURES		TOTAL
HI D	2.432	17.528	1.236	34.472	52
		11		41	
LOW D	3.417	12.472	1.737	24.528	37
		19		18	
TOTAL	30		59		89

$$Q = 2.432 + 3.417 + 1.236 + 1.737 = 8.822$$

APPENDIX B: INTERACTIVE EFFECTS

TABLE B-1: AC INTERACTION

EFFECT	SUCCESSES		FAILURES		TOTAL
HI A/HI C LOW A/LOW C	2.083	14.494	1.059	28.506	43
		9		34	
HI A/LOW C LOW A/HI C	1.947	15.506	0.990	30.494	46
		21		25	
TOTAL	30		59		89

$Q = 2.083 + 1.947 + 1.059 + .990 = 6.079$

TABLE B-2: AD INTERACTION

EFFECT	SUCCESSES		FAILURES		TOTAL
HI A/HI D LOW A/LOW D	1.401	14.494	0.712	28.506	43
		19		24	
HI A/LOW D LOW A/HI D	1.309	15.506	0.666	30.494	46
		11		35	
TOTAL	30		59		89

$Q = 1.401 + 1.309 + .712 + .666 = 4.088$

TABLE B-3: BC INTERACTION

EFFECTS	SUCCESSES		FAILURES		TOTAL
HI B/HI C LOW B/LOW C	0.317	14.831	0.161	29.169	44
		17		27	
HI B/LOW C LOW B/HI C	0.310	15.169	0.158	29.831	45
		13		32	
TOTAL	30		59		89

$$Q = .317 + .310 + .161 + .158 = 0.946$$

TABLE B-4: BD INTERACTION

EFFECT	SUCCESSES		FAILURES		TOTAL
HI B/HI D LOW B/LOW D	2.173	12.135	1.105	23.865	36
		7		29	
HI B/LOW D LOW B/HI D	1.476	17.865	0.750	35.135	53
		23		30	
TOTAL	30		59		89

$$Q = 2.173 + 1.476 + 1.105 + .750 = 5.504$$

TABLE B-5: CD INTERACTION

EFFECT	SUCCESSES		FAILURES		TOTAL
HI C/HI D LOW C/LOW D	0.187	12.472	0.095	24.528	37
		14		23	
HI C/LOW D LOW C/HI D	0.133	17.528	0.068	34.472	52
		16		36	
TOTAL	30		59		89

$$Q = .187 + .133 + .095 + .068 = 0.483$$

APPENDIX C: ANOVA PROGRAM

```c
 #include <stdio.h>
 #include <iostream.h>
 #include <math.h>
 #include <conio.h>
 #include <string.h>
/* Function: factorial of a number */

double factorial(float start)
{
 float m=start;
 double Tl =m;
 if (Tl == 0.0) Tl = 1;
 while (m > 2.0)
 {
    Tl *= (m-1);
    --m;
 }
 if (m==.5)
     return(0.8862269255);
 if (m==1.5)
     return(Tl * 0.8862269255);
 else
     return(Tl);
}

   /*    Function Fdistn        */

double Fdistn(int Fl, int F2, float x)
{
 int i,H,S;
 float M[70],N[70],Q[70];
 int D[2]={Fl,F2};
 float A=0.0,B=0.0,C=0.0,J=0.0,K=0.0,T=0.0,Z=0.0;
 C = x;
 if ((D[0] == 1.0) || (D[1] == 1.0))
 {
  if (D[0] < D[1])
   H = D[1];
  else
   H = D[0];
```

```
if (H>1)
{
 if ((H%2) == 0.0)
 {
  if (D[0] == 1.0)
   S=1.0;
  else
   S=2.0;
  J = (S-1.0);
  C = (1.0/(1.0+(C*D[0]/D[1])));
  C *= pow(-1.0, J);
  C += J;
  K = (H-2.0)/2.0;
  for (i=0; i<K; ++i)
  {
   M[i] = (2.0*i)+2.0;
   M[i] = (M[i]-1.0)/M[i];
  }
  N[0] = 1.0;
  for (i=0; i<(K-1); ++i)
  {
   M[i+1] *= M[i];
   N[i+1] = M[i];
  }
  N[K] = M[K-1];
  for (i=0; i<=K; ++i)
  {
   M[i] = i;
   M[i] = pow(C, M[i]);
   M[i] *= N[i];
   T += M[i];
  }
  T *= pow((1.0-C),0.5);
  T *= pow(-1.0,J);
  return(T+J);
 }
 else
 {
  if (H == D[0])
   S=1;
  else
   S=2;
  J = pow((C*D[0]/D[1]),0.5);
  J = atan(J);
  K = (H-3.0)/2.0;
```

```c
      if (S==1)
       B = sin(J);
      else
       B = cos(J);
      for (i=0; i<K; ++i)
      {
       N[i] = M[i] = (i+1.0)*2.0;
       N[i] = pow(B,N[i]);
       M[i] = M[i]/(M[i]+1.0);
      }
      for (i=1; i<K; ++i)
      {
       M[i] *= M[i-1];
      }
      B = 1.0;
      for (i=0; i<K; ++i)
      {
       B += (M[i]*N[i]);
      }
      B = (B*cos(J)*sin(J)*pow(-1.0,S))+J;
      return(2.0*B/3.141592654);
     }
    }
    else
     return(1.0-(2.0*atan(1.0/(pow(C,0.5))))/3.141592654));
  }
  else
  {
   if (((D[0]%2) == 0.0) || ((D[1]%2) == 0.0))
     {
     if ((((D[0]%2) == 0.0) && ((D[1]%2) == 0.0))
        {
           if (D[0] > D[1])
           S=2.0;
           else
           S=1.0;
        }
     else
        {
           if ((D[0]%2)== 0.0)
           S=1.0;
           else
           S=2.0;
        }
     C=(pow(-1.0,(S-1.0)))*(1.0/(1.0+(C*D[0]*1.0/D[1])));
     C += (S-1.0);
```

```
        if (S == 1.0)
            T = 2;
        else
            T = 1;
        if (D[S-1] < 4.0)
            Z=pow(x,(D[T-1])/2.0);
        else
            {
                N[0] = D[T-1];
                B=1.0;
                A=(D[S-1]-2.0)/2.0;
                for (i=0; i<A; ++i)
                M[i] = 2.0*(i+1);
                A=(D[S-1]-4.0)/2.0;
                for (i=1; i<=A; ++i)
              {
               N[i] = 2.0*(i);
               N[i] += D[T-1];
              }
                for (i=0; i<=A; ++i)
              {
               N[i] /= M[i];
               M[i] = pow((1.0-C),M[i]/2.0);
              }
                for (i=0; i<A; ++i)
                N[i+1] *= N[i];
                for (i=0; i<=A; ++i)
              {
               M[i] *= N[i];
               B += M[i];
              }
                Z = (B*pow(C,(D[T-1]/2.0)));
            }
        B = (S%2);
        C = (pow((1.0-Z),B)*pow(Z,(1.0-B)));
        return(C);
        }
    else
        {
    A = ((D[1]-3)/2.0);
    N[0] = 1;
    for (i=0; i<A; ++i)
      {
        M[i] = 2*(i+1);
        M[i] /= M[i]+1;
        N[i+1] = M[i];
        N[i+1] *= N[i];
      }
```

34

```c
    T = atan(pow((C*D[0])/D[1],0.5));
    J=0.0;
    for (i=0; i<=A; ++i)
      {
        M[i] = (2*i)+1;
        M[i] = pow(cos(T),M[i]);
        M[i] *= N[i];
        J += M[i];
      }
    A = ((J*sin(T))+T)*2.0/3.141592654;
    B = ((D[0]-3)/2.0);
    Z = 1.0;
    for (i=0; i<B; ++i)
      {
        N[i] = M[i] = 2*(i+1);
        M[i] = (M[i]+D[1]-1)/(M[i]+1);
      }
    for (i=0; i<B; ++i)
      {
        M[i+1] *= M[i];
        N[i] = pow((sin(T)),N[i]);
        M[i] *= N[i];
        Z += M[i];
      }
    Z *= (sin(T))*pow(cos(T),D[1]);
    K = factorial((D[1]-2)/2.0);
    K = 1.0/K;
    Z *= K;
    K = factorial((D[1]-1)/2.0);
    Z *= K;
    Z *= (2.0/(pow(3.141592654,0.5)));
    return(A-Z);
     }
  }
 }


/*    Function Stu    */
double STU(int d, float p)
  {
    float z;
    z = Fdistn(1,d,(pow(p,2)));
    if (p>0.0)
        return ((z+1)/2.0);
    else
        return(((-1.0*z)+1.0)/2.0);
  }
```

```c
/*   Function NQUAN      */
double Nquan(float p)
{
 int n;
 float Q=(p-0.5);
 float A1[4]={2.5066282388, -18.615000625, 41.3911977353,
-25.4410604964};
 float B1[4]={-8.473510931, 23.0833674374, -21.0622410183,
3.1308290983};
 float A2[4]={-2.7871893114, -2.2979647913, 4.8501412714,
2.3212127686};
 float B2[2]={3.5438892476, 1.63706781897};
 float C[4]={1,2,3,4};
 float F[4];
 float E=1.0,G=0.0,T=0.0;
 if (fabs(Q) <= 0.42)
 {
  T=Q;
  Q=T*T;
  for (n=0;n<4;++n)
  {
   F[n]=pow(Q, C[n]);
   B1[n] *= F[n];
   E += B1[n];
   --C[n];
   F[n]=pow(Q, C[n]);
   A1[n] *= F[n];
   G += A1[n];
  }
  T *= (G/E);
  return(T);
 }
 else
 {
  T=pow(fabs(log(0.5-(fabs(Q)))),0.5);
  for (n=0;n<2;++n)
  {
   C[n] = pow(T,(n+1));
   C[n] *= B2[n];
   E += C[n];
  }
  for (n=0;n<4;++n)
  {
   A2[n] *= pow(T,n);
   G += A2[n];
  }
```

```
  if (Q<0)
   return(-1*G/E);
  else
   return(G/E);
 }
}

/*  Function Tquan     ****/

double Tquan(int d,float p)
{
 int n,y;
 double z=0.0,x=0.0,DENS=0.0;
 float T1=0.0,T2=0.0,V1=0.0,V2=0.0;
 float TA[5]={0,0,0,0,0};
 printf("d=%d.\n",&d);
 if (d==1.0) z=tan((p-0.5)*3.141592654);
 else
 {
  if (d==2.0)
   z=((2.*p)-1.)/pow((2.*p*(1-p)), 0.5);
  else
  {
   x=Nquan(p);
   TA[0] = x;
   TA[1] = (TA[.] + pow(x,3.0))/4.0;
   T    A    [    2    ]              =
(x/32.0)+((pow(x,3.0))/6.0)+((pow(x,5.0))*5.0/96.0);
   T    A    [    3    ]              =
(x*-15.0/384.0)+((pow(x,3.0))*17.0/384.0)+((pow(x,5.0))*19.0
/384.0)
       +((pow(x,7.0))/128.0);
   T    A    [    4    ]              =
(x*-945.0/92160.0)+((pow(x,3.0))/48.0)+((pow(x,5.0))*1482.0/
92160.0)

+((pow(x,7.0))*776.0/92160.0)+((pow(x,9.0))*79.0/92160.0);
   for (n=0;n<5;++n)
   {
    TA[n] *= pow(d, -n);
    z += TA[n];
   }
   if (d>37.0)
   y=2;
```

```c
      else
      {
       if (d<6.0)
       y=4;
       else
       y=3;
      }
      T1 = (d-1.0)/2.0;
      T2 = (d-2.0)/2.0;
      T1 = factorial(T1);
      T2 = factorial(T2);
      for (n=1; n<=y; ++n)
      {
       V1 = 1.0 + ((z*z)/d);
       V1 = pow(V1,((d+1.0)/2));
       V2 = pow((d*3.141592654),0.5);
       DENS = T1/(T2*V1*V2);
       z -= (STU(d,z)-p)/DENS;
      }
     }
    }
   }
  return(z);
}
/*  Function  Fquan  */

double Fquan(int d1, int d2, float p)
{
   int i,D[2];
   float A=0.0,B=0.0,N=0.0,SUM=0.0,z=0.0;
   double L = 0.0;
   float U[2];
   D[0] = d1;
   D[1] = d2;
   if ((D[0] == 1) || (D[1] == 1))
      {
   if (D[0] == 1)
      L=1.0;
   else
      L=2.0;
   if (D[0] < D[1])
      A = D[1];
   else
      A = D[0];
   B = pow(-1.0,L+1.0);
   B = ((B*p)+L)/2.0;
   z = Tquan(A,B);
   z *= z;
```

38

```
          if (L == 2.0)
            return(1.0/z);
        else
            return(z);
          }
        else
          {
        if ((D[0] == 2) || (D[1] == 2))
            {
              if (D[0] < D[1])
                  L = D[1];
              else
                  L = D[0];
              N = 2.0/L;
              A = D[0]*1.0/D[1];
              if (A >= 1.0)
                  {
                  B = p;
                  z = (D[1]*1.0/D[0])*((pow(B,N))/(1.0-pow(B,N)));
                  return(z);
                  }
              else
                  {
                  B = 1.0-p;
                  z = (D[1]/D[0])/((pow(B,N))/(1.0-pow(B,N)));
                  return(z);
                  }

            }
        else
            {
            A = Nquan(1.0-p);
            U[0] = D[1]-1.0;
            U[1] = D[0]-1.0;
            B = (1.0/U[0])+(1.0/U[1]);
            N = (1.0/U[0])-(1.0/U[1]);
            L = (pow(A,2)-3.0)/6.0;
            SUM = A*(pow(A,2)+11.0)/144.0;
            SUM *= (pow((2.0/B),0.5))*pow(N,2.0);
            SUM = ((5.0/6.0)-(B/3.0))+L-SUM;
            SUM *= N;
            z = ((pow(((2.0/B)+L),0.5))*A*B/2.0)-SUM;
            z = exp(-2.0*z);
            i = 1;
            do
                {
              SUM = (D[0]+D[1])/2.0;
              SUM = pow(((z*D[0]/D[1])+1.0),SUM);
```

```
              L = pow(z,((D[0]/2.0)-1.0));
              L = L/SUM;
              B = D[0]*1.0/D[1]*1.0;
              L *= pow(B,(D[0]/2.0));
              N                                              =
(factorial((D[0]/2.0)-1.0))*factorial((D[1]/2.0)-1.0);
              A = (factorial(((D[0]+D[1])/2.0)-1.0)/N)*L;
              z =  z - ((Fdistn(D[0],D[1],z)-p)/A);
              ++i;
                }
            while (i<=3);
            return(z);


          }
          }
    }


    void main()
    {
     double test,anova;
     int d1,d2,i,j,k,n,l,DOFr,DOFa,DOFb,opt,in1,in2,num;
     float N[33][20], S[32];
     float x,y,p,q,Total,Total2,Ave;
     float SSr,SSt,SSa,SSb,start,Fa,Fb,MSa,MSb,MSr;
     printf("Welcome  to  the  ANOVA  procedure.   If  you  are
unfamiliar with\n");
     printf("how ANOVA works, you should review its restrictions
first.\n");
     printf("Make   sure   your   data   comes   from   normal
distributions.  For \n");
     printf("factorial  problems,  the  data  must  be  entered  in
standard order.\n");
     printf("\n\nEnter (1) for One-way ANOVA\n");
     printf("Enter (2) for Two-way ANOVA\n");
     printf("Enter (3) for Factorial ANOVA\n");
     scanf("%d",&opt);
     if (opt==1)
       {
         SSt = 0.0;
         SSa = 0.0;
         num = 0;
         printf("Enter number of effects (limit 10)\n");
         scanf("%d",&in1);
         Total = 0.0;
         for (i=1;i<=in1;++i)
       {
         S[i-1] = 0.0;
```

40

```c
        printf("Enter   number   of   data   points   for   effect
%d:\n",i);
        scanf("%d",&in2);
        printf("Press enter after each entry.\n");
        for (j=1;j<=in2;++j)
          {
             scanf("%g",&x);
             SSt += pow(x,2);
             Total += x;
             num += 1.0;
             S[i-1] += x;
          }
        SSa += pow(S[i-1],2)/in2;

    }
    SSt = SSt - pow(Total,2)/num;
    SSa -= pow(Total,2)/num;
    SSb = SSt - SSa;
    DOFa = in1 - 1;
    DOFr = num - 1;
    DOFb = DOFr - DOFa;
    printf("Enter significance level: ");
    fflush(stdin);
    scanf("%g", &p);
    Fa = (SSa/DOFa)/(SSb/DOFb);
    MSa = (SSa/DOFa);
    MSb = (SSb/DOFb);
    printf("   EFFECT             SS          DOF           MS");
    printf("            F         SIGNIF\n");
    printf("   ------            --           ---           --");
    printf("        -           ------\n");
    printf("   Between\n");
    printf("populations         %g            %d          %g
%g",SSa,DOFa,MSa,Fa);
    test=Fquan(DOFa,DOFb,p);
    if (test < Fa)
      printf("      YES\n");
    else
      printf("       NO\n");
    printf("Residual                 %g                %d
%g\n",SSb,DOFb,MSb);
    printf("Total           %g         %d\n",SSt,DOFr);
  }
  if (opt==2)
    {
       SSt = 0.0;
       SSa = 0.0;
       SSb = 0.0;
       y = 0.0;
```

```c
        num = 0;
        printf("Enter number of effects (limit 10)\n");
        scanf("%d",&in1);
        printf("Enter number of blocks (limit 10)\n");
        scanf("%d",&in2);
        Total = 0.0;
        printf("Enter data by column.  Press Enter after each
point.\n");
        for (i=1;i<=in1;++i)
        {
          Total = 0.0;
          for (j=1;j<=in2;++j)
            {
              scanf("%g",&x);
              N[i][j] = x;
              Total += x;
              y += x;
              num += 1.0;
            }
          N[i][in2+1] = Total/in2;
        }
        y /=num;

        for (j=1;j<=in2;++j)
          {
            N[in1 + 1][j] = 0.0;
            for (i=1;i<=in1;++i)
              {
               N[in1 + 1][j] += N[i][j];
               SSt += pow((N[i][j] - y),2);
               }
            N[in1 + 1][j] /= in1;
          }
        for (i=1;i<=in2;++i)
          {
            if (i<=in1)  SSa += pow((N[i][in2 + 1] - y),2);
            SSb += pow((N[in1 + 1][i] - y),2);
          }
        SSa *= in2;
        SSb *= in1;
        SSr = SSt - SSa - SSb;
        printf("Enter significance level: ");
        fflush(stdin);
        scanf("%g", &p);
        DOFa = in1 - 1;
        DOFb = in2 - 1;
        DOFr = DOFa*DOFb;
        Fa = (SSa/DOFa)/(SSr/DOFr);
        Fb = (SSb/DOFb)/(SSr/DOFr);
```

```c
    MSa = (SSa/DOFa);
    MSb = (SSb/DOFb);
    MSr = (SSr/DOFr);
    printf("  EFFECT            SS          DOF          MS");
    printf("              F          SIGNIF\n");
    printf("  ------             --         ---          --");
    printf("        -          ------\n");
    printf("    Among\n");
    printf("populations        %g           %d          %g
%g",SSa,DOFa,MSa,Fa);
    test=Fquan(DOFa,DOFr,p);
    if (test < Fa)
      printf("       YES\n");
    else
      printf("        NO\n");
    printf("Blocks              %g           %d          %g
%g",SSb,DOFb,MSb,Fb);
    test=Fquan(DOFb,DOFr,p);
    if (test < Fb)
      printf("       YES\n");
    else
      printf("        NO\n");
    printf("Residual                 %g               %d
%g\n",SSr,DOFr,MSr);
    printf("Total            %g         %d\n",SSt,(num-1));
    }
  else
  {
  struct comp
    {
      char name[6];
      float sum;
    } SS[32];
  char *str1 = "A";
  strcpy(SS[0].name, str1);
  char *str2 = "B";
  strcpy(SS[1].name, str2);
  char *str3 = "AB";
  strcpy(SS[2].name, str3);
  char *str4 = "C";
  strcpy(SS[3].name, str4);
  char *str5 = "AC";
  strcpy(SS[4].name, str5);
  char *str6 = "BC";
  strcpy(SS[5].name, str6);
  char *str7 = "ABC";
  strcpy(SS[6].name, str7);
  char *str8 = "D";
  strcpy(SS[7].name, str8);
```

43

```
char *str9 = "AD";
strcpy(SS[8].name, str9);
char *str10= "BD";
strcpy(SS[9].name, str10);
char *str11 = "CD";
strcpy(SS[10].name, str11);
char *str12 = "ABD";
strcpy(SS[11].name, str12);
char *str13 = "ACD";
strcpy(SS[12].name, str13);
char *str14 = "BCD";
strcpy(SS[13].name, str14);
char *str15 = "ABCD";
strcpy(SS[14].name, str15);
char *str16 = "E";
strcpy(SS[15].name, str16);
char *str17 = "AE";
strcpy(SS[16].name, str17);
char *str18 = "BE";
strcpy(SS[17].name, str18);
char *str19 = "CE";
strcpy(SS[18].name, str19);
char *str20 = "DE";
strcpy(SS[19].name, str20);
char *str21 = "ABE";
strcpy(SS[20].name, str21);
char *str22 = "ACE";
strcpy(SS[21].name, str22);
char *str23 = "ADE";
strcpy(SS[22].name, str23);
char *str24 = "BCE";
strcpy(SS[23].name, str24);
char *str25 = "BDE";
strcpy(SS[24].name, str25);
char *str26 = "CDE";
strcpy(SS[25].name, str26);
char *str27 = "ABCE";
strcpy(SS[26].name, str27);
char *str28 = "ABDE";
strcpy(SS[27].name, str28);
char *str29 = "ACDE";
strcpy(SS[28].name, str29);
char *str30 = "BCDE";
strcpy(SS[29].name, str30);
char *str31 = "ABCDE";
strcpy(SS[30].name, str31);
printf("Enter the number of effects: ");
fflush(stdin);
scanf("%d", &k);
```

```c
    l = pow(2,k)-1;
    i=0;
    j=0;
    Total = 0.0;
    printf("Enter the number of replications:  ");
    scanf("%d", &n);
    y = pow(2,k)*n;
    SSt = 0.0;
    printf("Enter data in standard order.  Press enter\n");
    printf("after each entry.\n");
    for (i=0;i<=l;++i)
      {
        S[i] = 0.0;
        for (j=0;j<=(n-1);++j)
        {

          scanf("%g",&x);
          N[i][j] = x;
          Total += N[i][j];
          S[i] += x;
      }
    }
    Ave = Total/(pow(2,k)*n);
    if (k==2)
      {
        SS[0].sum = (1/y)*pow((S[1]+S[3]-S[0]-S[2]),2);
        SS[1].sum = (1/y)*pow((S[2]+S[3]-S[0]-S[1]),2);
        SS[2].sum = (1/y)*pow((S[0]+S[3]-S[1]-S[2]),2);
        for (i=0;i<=l;++i)
        {
          for (j=0;j<=(n-1);++j)
            {
              SSt += pow((N[i][j] - Ave),2);
            }
      }
        SSr = SSt - SS[0].sum - SS[1].sum - SS[2].sum;
      }
    if (k==3)
      {
                        S S [ 0 ] . s u m       =
(1/y)*pow((S[1]+S[3]+S[5]+S[7]-S[0]-S[2]-S[4]-S[6]),2);
                        S S [ 1 ] . s u m       =
(1/y)*pow((S[2]+S[3]+S[6]+S[7]-S[0]-S[1]-S[4]-S[5]),2);
                        S S [ 2 ] . s u m       =
(1/y)*pow((S[0]+S[3]+S[4]+S[7]-S[1]-S[2]-S[5]-S[6]),2);
                        S S [ 3 ] . s u m       =
(1/y)*pow((S[4]+S[5]+S[6]+S[7]-S[0]-S[1]-S[2]-S[3]),2);
                        S S [ 4 ] . s u m       =
(1/y)*pow((S[0]+S[2]+S[5]+S[7]-S[1]-S[3]-S[4]-S[6]),2);
```

45

```
                              S  S  [  5  ]  .  s  u  m       =
(1/y)*pow((S[0]+S[1]+S[6]+S[7]-S[2]-S[3]-S[4]-S[5]),2);
                              S  S  [  6  ]  .  s  u  m       =
(1/y)*pow((S[1]+S[2]+S[4]+S[7]-S[0]-S[3]-S[5]-S[6]),2);
      for (i=0;i<=1;++i)
    {
       for (j=0;j<=(n-1);++j)
         {
            SSt += pow((N[i][j] - Ave),2);
         }
    }
     SSr = SSt - SS[0].sum - SS[1].sum - SS[2].sum -
SS[3].sum - SS[4].sum
            - SS[5].sum - SS[6].sum;
    }
   if (k==4)
     {
                              S  S  [  0  ]  .  s  u  m       =
(1/y)*pow((S[1]+S[3]+S[5]+S[7]+S[9]+S[11]+S[13]+S[15]
            -S[0]-S[2]-S[4]-S[6]-S[8]-S[10]-S[12]-S[14]),2);
                              S  S  [  1  ]  .  s  u  m       =
(1/y)*pow((S[2]+S[3]+S[6]+S[7]+S[10]+S[11]+S[14]+S[15]
            -S[0]-S[1]-S[4]-S[5]-S[8]-S[9]-S[12]-S[13]),2);
                              S  S  [  2  ]  .  s  u  m       =
(1/y)*pow((S[0]+S[3]+S[4]+S[7]+S[8]+S[11]+S[12]+S[15]
            -S[1]-S[2]-S[5]-S[6]-S[9]-S[10]-S[13]-S[14]),2);
                              S  S  [  3  ]  .  s  u  m       =
(1/y)*pow((S[4]+S[5]+S[6]+S[7]+S[12]+S[13]+S[14]+S[15]
            -S[0]-S[1]-S[2]-S[3]-S[8]-S[9]-S[10]-S[11]),2);
                              S  S  [  4  ]  .  s  u  m       =
(1/y)*pow((S[0]+S[2]+S[5]+S[7]+S[8]+S[10]+S[13]+S[15]
            -S[1]-S[3]-S[4]-S[6]-S[9]-S[11]-S[12]-S[14]),2);
                              S  S  [  5  ]  .  s  u  m       =
(1/y)*pow((S[0]+S[1]+S[6]+S[7]+S[8]+S[9]+S[14]+S[15]
            -S[2]-S[3]-S[4]-S[5]-S[10]-S[11]-S[12]-S[13]),2);
                              S  S  [  6  ]  .  s  u  m       =
(1/y)*pow((S[1]+S[2]+S[4]+S[7]+S[9]+S[10]+S[12]+S[15]
            -S[0]-S[3]-S[5]-S[6]-S[8]-S[11]-S[13]-S[14]),2);
                              S  S  [  7  ]  .  s  u  m       =
(1/y)*pow((S[8]+S[9]+S[10]+S[11]+S[12]+S[13]+S[14]+S[15]
            -S[0]-S[1]-S[2]-S[3]-S[4]-S[5]-S[6]-S[7]),2);
                              S  S  [  8  ]  .  s  u  m       =
(1/y)*pow((S[0]+S[2]+S[4]+S[6]+S[9]+S[11]+S[13]+S[15]
            -S[1]-S[3]-S[5]-S[7]-S[8]-S[10]-S[12]-S[14]),2);
                              S  S  [  9  ]  .  s  u  m       =
(1/y)*pow((S[0]+S[1]+S[4]+S[5]+S[10]+S[11]+S[14]+S[15]
            -S[2]-S[3]-S[6]-S[7]-S[8]-S[9]-S[12]-S[13]),2);
```

```
                       S  S  [   1  0   ]   .   s   u   m         =
(1/y)*pow((S[0]+S[1]+S[2]+S[3]+S[12]+S[13]+S[14]+S[15]
         -S[4]-S[5]-S[6]-S[7]-S[8]-S[9]-S[10]-S[11]),2);
                       S  S  [   1  1   ]   .   s   u   m         =
(1/y)*pow((S[1]+S[2]+S[5]+S[6]+S[8]+S[11]+S[12]+S[15]
         -S[0]-S[3]-S[4]-S[7]-S[9]-S[10]-S[13]-S[14]),2);
                       S  S  [   1  2   ]   .   s   u   m         =
(1/y)*pow((S[1]+S[3]+S[4]+S[6]+S[8]+S[10]+S[13]+S[15]
         -S[0]-S[2]-S[5]-S[7]-S[9]-S[11]-S[12]-S[14]),2);
                       S  S  [   1  3   ]   .   s   u   m         =
(1/y)*pow((S[2]+S[3]+S[4]+S[5]+S[8]+S[9]+S[14]+S[15]
         -S[0]-S[1]-S[6]-S[7]-S[10]-S[11]-S[12]-S[13]),2);
                       S  S  [   1  4   ]   .   s   u   m         =
(1/y)*pow((S[0]+S[3]+S[5]+S[6]+S[9]+S[10]+S[12]+S[15]
         -S[1]-S[2]-S[4]-S[7]-S[8]-S[11]-S[13]-S[14]),2);
       for (i=0;i<=1;++i)
      {
        for (j=0;j<=(n-1);++j)
          {
            SSt += pow((N[i][j] - Ave),2);
          }
      }
       SSr = SSt - SS[0].sum - SS[1].sum - SS[2].sum -
SS[3].sum - SS[4].sum
          - SS[5].sum - SS[6].sum - SS[7].sum - SS[8].sum -
SS[9].sum
          - SS[10].sum - SS[11].sum - SS[12].sum - SS[13].sum
- SS[14].sum;
      }
    if (k==5)
      {
                       S  S  [   0   ]   .   s   u   m         =
(1/y)*pow((S[1]+S[3]+S[5]+S[7]+S[9]+S[11]+S[13]+S[15]
         +S[17]+S[19]+S[21]+S[23]+S[25]+S[27]+S[29]+S[31]
         -S[0]-S[2]-S[4]-S[6]-S[8]-S[10]-S[12]-S[14]

-S[16]-S[18]-S[20]-S[22]-S[24]-S[26]-S[28]-S[30]),2);
                       S  S  [   1   ]   .   s   u   m         =
(1/y)*pow((S[2]+S[3]+S[6]+S[7]+S[10]+S[11]+S[14]+S[15]
         +S[18]+S[19]+S[22]+S[23]+S[26]+S[27]+S[30]+S[31]
         -S[0]-S[1]-S[4]-S[5]-S[8]-S[9]-S[12]-S[13]
      -S[16]-S[17]-S[20]-S[21]-S[24]-S[25]-S[28]-S[29]),2);
```

```
                          S   S   [   2   ]   .   s   u   m           =
(1/y)*pow((S[0]+S[3]+S[4]+S[7]+S[8]+S[11]+S[12]+S[15]
          +S[17]+S[19]+S[20]+S[23]+S[24]+S[27]+S[28]+S[31]
          -S[1]-S[2]-S[5]-S[6]-S[9]-S[10]-S[13]-S[14]
      -S[16]-S[18]-S[21]-S[22]-S[25]-S[26]-S[29]-S[30]),2);
                          S   S   [   3   ]   .   s   u   m           =
(1/y)*pow((S[4]+S[5]+S[6]+S[7]+S[12]+S[13]+S[14]+S[15]
          +S[20]+S[21]+S[22]+S[23]+S[28]+S[29]+S[30]+S[31]
          -S[0]-S[1]-S[2]-S[3]-S[8]-S[9]-S[10]-S[11]
      -S[16]-S[17]-S[18]-S[19]-S[24]-S[25]-S[26]-S[27]),2);
                          S   S   [   4   ]   .   s   u   m           =
(1/y)*pow((S[0]+S[2]+S[5]+S[7]+S[8]+S[10]+S[13]+S[15]
          +S[16]+S[18]+S[21]+S[23]+S[24]+S[26]+S[29]+S[31]
          -S[1]-S[3]-S[4]-S[6]-S[9]-S[11]-S[12]-S[14]
      -S[17]-S[19]-S[20]-S[22]-S[25]-S[27]-S[28]-S[30]),2);
                          S   S   [   5   ]   .   s   u   m           =
(1/y)*pow((S[0]+S[1]+S[6]+S[7]+S[8]+S[9]+S[14]+S[15]
          +S[16]+S[17]+S[22]+S[23]+S[24]+S[25]+S[30]+S[31]
          -S[2]-S[3]-S[4]-S[5]-S[10]-S[11]-S[12]-S[13]
      -S[18]-S[19]-S[20]-S[21]-S[26]-S[27]-S[28]-S[29]),2);
                          S   S   [   6   ]   .   s   u   m           =
(1/y)*pow((S[1]+S[2]+S[4]+S[7]+S[9]+S[10]+S[12]+S[15]
          +S[17]+S[18]+S[20]+S[23]+S[25]+S[26]+S[28]+S[31]
          -S[0]-S[3]-S[5]-S[6]-S[8]-S[11]-S[13]-S[14]
      -S[16]-S[19]-S[21]-S[22]-S[24]-S[27]-S[29]-S[30]),2);
                          S   S   [   7   ]   .   s   u   m           =
(1/y)*pow((S[8]+S[9]+S[10]+S[11]+S[12]+S[13]+S[14]+S[15]
          +S[24]+S[25]+S[26]+S[27]+S[28]+S[29]+S[30]+S[31]
          -S[0]-S[1]-S[2]-S[3]-S[4]-S[5]-S[6]-S[7]
      -S[16]-S[17]-S[18]-S[19]-S[20]-S[21]-S[22]-S[23]),2);
                          S   S   [   8   ]   .   s   u   m           =
(1/y)*pow((S[0]+S[2]+S[4]+S[6]+S[9]+S[11]+S[13]+S[15]
          +S[16]+S[18]+S[20]+S[22]+S[25]+S[27]+S[29]+S[31]
          -S[1]-S[3]-S[5]-S[7]-S[8]-S[10]-S[12]-S[14]
      -S[17]-S[19]-S[21]-S[23]-S[24]-S[26]-S[28]-S[30]),2);
                          S   S   [   9   ]   .   s   u   m           =
(1/y)*pow((S[0]+S[1]+S[4]+S[5]+S[10]+S[11]+S[14]+S[15]
          +S[16]+S[17]+S[20]+S[21]+S[26]+S[27]+S[30]+S[31]
          -S[2]-S[3]-S[6]-S[7]-S[8]-S[9]-S[12]-S[13]
      -S[18]-S[19]-S[22]-S[23]-S[24]-S[25]-S[28]-S[29]),2);
                          S   S   [   1   0   ]   .   s   u   m           =
(1/y)*pow((S[0]+S[1]+S[2]+S[3]+S[12]+S[13]+S[14]+S[15]
         +S[16]+S[17]+S[18]+S[19]+S[28]+S[29]+S[30]+S[31]
         -S[4]-S[5]-S[6]-S[7]-S[8]-S[9]-S[10]-S[11]

-S[20]-S[21]-S[22]-S[23]-S[24]-S[25]-S[26]-S[27]),2);
```

48

```
                    S  S  [  1  1  ]  .  s  u  m       =
(1/y)*pow((S[1]+S[2]+S[5]+S[6]+S[8]+S[11]+S[12]+S[15]
        +S[17]+S[18]+S[21]+S[22]+S[24]+S[27]+S[28]+S[31]
        -S[0]-S[3]-S[4]-S[7]-S[9]-S[10]-S[13]-S[14]

-S[16]-S[19]-S[20]-S[23]-S[25]-S[26]-S[29]-S[30]),2);
                    S  S  [  1  2  ]  .  s  u  m       =
(1/y)*pow((S[1]+S[3]+S[4]+S[6]+S[8]+S[10]+S[13]+S[15]
        +S[17]+S[19]+S[20]+S[22]+S[24]+S[26]+S[29]+S[31]
        -S[0]-S[2]-S[5]-S[7]-S[9]-S[11]-S[12]-S[14]

-S[16]-S[18]-S[21]-S[23]-S[25]-S[27]-S[28]-S[30]),2);
                    S  S  [  1  3  ]  .  s  u  m       =
(1/y)*pow((S[2]+S[3]+S[4]+S[5]+S[8]+S[9]+S[14]+S[15]
        +S[18]+S[19]+S[20]+S[21]+S[24]+S[25]+S[30]+S[31]
        -S[0]-S[1]-S[6]-S[7]-S[10]-S[11]-S[12]-S[13]

-S[16]-S[17]-S[22]-S[23]-S[26]-S[27]-S[28]-S[29]),2);
                    S  S  [  1  4  ]  .  s  u  m       =
(1/y)*pow((S[0]+S[3]+S[5]+S[6]+S[9]+S[10]+S[12]+S[15]
        +S[16]+S[19]+S[21]+S[22]+S[25]+S[26]+S[28]+S[31]
        -S[1]-S[2]-S[4]-S[7]-S[8]-S[11]-S[13]-S[14]

-S[17]-S[18]-S[20]-S[23]-S[24]-S[27]-S[29]-S[30]),2);
                    S  S  [  1  5  ]  .  s  u  m       =
(1/y)*pow((S[16]+S[17]+S[18]+S[19]+S[20]+S[21]+S[22]

+S[23]+S[24]+S[25]+S[26]+S[27]+S[28]+S[29]+S[30]+S[31]
        -S[0]-S[1]-S[2]-S[3]-S[4]-S[5]-S[6]-S[7]
        -S[8]-S[9]-S[10]-S[11]-S[12]-S[13]-S[14]-S[15]),2);
                    S  S  [  1  6  ]  .  s  u  m       =
(1/y)*pow((S[0]+S[2]+S[4]+S[6]+S[8]+S[10]+S[12]+S[14]
        +S[17]+S[19]+S[21]+S[23]+S[25]+S[27]+S[29]+S[31]
        -S[1]-S[3]-S[5]-S[7]-S[9]-S[11]-S[13]-S[15]

-S[16]-S[18]-S[20]-S[22]-S[24]-S[26]-S[28]-S[30]),2);
                    S  S  [  1  7  ]  .  s  u  m       =
(1/y)*pow((S[0]+S[1]+S[4]+S[5]+S[8]+S[9]+S[12]+S[13]
        +S[18]+S[19]+S[22]+S[23]+S[26]+S[27]+S[30]+S[31]
        -S[2]-S[3]-S[6]-S[7]-S[10]-S[11]-S[14]-S[15]

-S[16]-S[17]-S[20]-S[21]-S[24]-S[25]-S[28]-S[29]),2);
                    S  S  [  1  8  ]  .  s  u  m       =
(1/y)*pow((S[0]+S[1]+S[2]+S[3]+S[8]+S[9]+S[10]+S[11]
        +S[20]+S[21]+S[22]+S[23]+S[28]+S[29]+S[30]+S[31]
        -S[4]-S[5]-S[6]-S[7]-S[12]-S[13]-S[14]-S[15]

-S[16]-S[17]-S[18]-S[19]-S[24]-S[25]-S[26]-S[27]),2);
```

```
                        S   S   [   1   9   ]   .   s  u   m          =
(1/y)*pow((S[0]+S[1]+S[2]+S[3]+S[4]+S[5]+S[6]+S[7]
         +S[24]+S[25]+S[26]+S[27]+S[28]+S[29]+S[30]+S[31]
         -S[8]-S[9]-S[10]-S[11]-S[12]-S[13]-S[14]-S[15]

-S[16]-S[17]-S[18]-S[19]-S[20]-S[21]-S[22]-S[23]),2);
                        S   S   [   2   0   ]   .   s  u   m          =
(1/y)*pow((S[1]+S[2]+S[5]+S[6]+S[9]+S[10]+S[13]+S[14]
         +S[16]+S[19]+S[20]+S[23]+S[24]+S[27]+S[28]+S[31]
         -S[0]-S[3]-S[4]-S[7]-S[8]-S[11]-S[12]-S[15]

-S[17]-S[18]-S[21]-S[22]-S[25]-S[26]-S[29]-S[30]),2);
                        S   S   [   2   1   ]   .   s  u   m          =
(1/y)*pow((S[1]+S[3]+S[4]+S[6]+S[9]+S[11]+S[12]+S[14]
         +S[16]+S[18]+S[21]+S[23]+S[24]+S[26]+S[29]+S[31]
         -S[0]-S[2]-S[5]-S[7]-S[8]-S[10]-S[13]-S[15]

-S[17]-S[19]-S[20]-S[22]-S[25]-S[27]-S[28]-S[30]),2);
                        S   S   [   2   2   ]   .   s  u   m          =
(1/y)*pow((S[1]+S[3]+S[5]+S[7]+S[8]+S[10]+S[12]+S[14]
         +S[16]+S[18]+S[20]+S[22]+S[25]+S[27]+S[29]+S[31]
         -S[0]-S[2]-S[4]-S[6]-S[9]-S[11]-S[13]-S[15]

-S[17]-S[19]-S[21]-S[23]-S[24]-S[26]-S[28]-S[30]),2);
                        S   S   [   2   3   ]   .   s  u   m          =
(1/y)*pow((S[2]+S[3]+S[4]+S[5]+S[10]+S[11]+S[12]+S[13]
         +S[16]+S[17]+S[22]+S[23]+S[24]+S[25]+S[30]+S[31]
         -S[0]-S[1]-S[6]-S[7]-S[8]-S[9]-S[14]-S[15]

-S[18]-S[19]-S[20]-S[21]-S[26]-S[27]-S[28]-S[29]),2);
                        S   S   [   2   4   ]   .   s  u   m          =
(1/y)*pow((S[2]+S[3]+S[6]+S[7]+S[8]+S[9]+S[12]+S[13]
         +S[16]+S[17]+S[20]+S[21]+S[26]+S[27]+S[30]+S[31]
         -S[0]-S[1]-S[4]-S[5]-S[10]-S[11]-S[14]-S[15]

-S[18]-S[19]-S[22]-S[23]-S[24]-S[25]-S[28]-S[29]),2);
                        S   S   [   2   5   ]   .   s  u   m          =
(1/y)*pow((S[4]+S[5]+S[6]+S[7]+S[8]+S[9]+S[10]+S[11]
         +S[16]+S[17]+S[18]+S[19]+S[28]+S[29]+S[30]+S[31]
         -S[0]-S[1]-S[2]-S[3]-S[12]-S[13]-S[14]-S[15]

-S[20]-S[21]-S[22]-S[23]-S[24]-S[25]-S[26]-S[27]),2);
                        S   S   [   2   6   ]   .   s  u   m          =
(1/y)*pow((S[0]+S[3]+S[5]+S[6]+S[8]+S[11]+S[13]+S[14]
         +S[17]+S[18]+S[20]+S[23]+S[25]+S[26]+S[29]+S[31]
         -S[1]-S[2]-S[4]-S[7]-S[9]-S[10]-S[12]-S[15]

-S[16]-S[19]-S[21]-S[22]-S[24]-S[27]-S[28]-S[30]),2);
```

50

```c
                              SS[27].sum         =
(1/y)*pow((S[0]+S[3]+S[4]+S[7]+S[9]+S[10]+S[13]+S[14]
        +S[17]+S[18]+S[21]+S[22]+S[24]+S[27]+S[28]+S[31]
        -S[1]-S[2]-S[5]-S[6]-S[8]-S[11]-S[12]-S[15]

-S[16]-S[19]-S[20]-S[23]-S[25]-S[26]-S[29]-S[30]),2);
                              SS[28].sum         =
(1/y)*pow((S[0]+S[2]+S[5]+S[7]+S[9]+S[11]+S[12]+S[14]
        +S[17]+S[19]+S[20]+S[22]+S[24]+S[26]+S[29]+S[31]
        -S[1]-S[3]-S[4]-S[6]-S[8]-S[10]-S[13]-S[15]

-S[30]-S[16]-S[18]-S[21]-S[23]-S[25]-S[27]-S[28]),2);
                              SS[29].sum         =
(1/y)*pow((S[0]+S[1]+S[6]+S[7]+S[10]+S[11]+S[12]+S[13]
        +S[18]+S[19]+S[20]+S[21]+S[24]+S[25]+S[30]+S[31]
        -S[2]-S[3]-S[4]-S[5]-S[8]-S[9]-S[14]-S[15]

-S[16]-S[17]-S[22]-S[23]-S[26]-S[27]-S[28]-S[29]),2);
                              SS[30].sum         =
(1/y)*pow((S[1]+S[2]+S[4]+S[7]+S[8]+S[11]+S[13]+S[14]
        +S[16]+S[19]+S[21]+S[22]+S[25]+S[26]+S[28]+S[31]
        -S[0]-S[3]-S[5]-S[6]-S[9]-S[10]-S[12]-S[15]

-S[17]-S[18]-S[20]-S[23]-S[24]-S[27]-S[29]-S[30]),2);

        for (i=0;i<=1;++i)
      {
        for (j=0;j<=(n-1);++j)
          {
            SSt += pow((N[i][j] - Ave),2);
          }
      }
       SSr = SSt;
       for (i=0;i<=30;++i)
      {
        SSr -= SS[i].sum;
      }
     }
  DOFr = y - 1 - 1;

  if (DOFr==0) DOFr=1;
  printf("Enter significance level: ");
  fflush(stdin);
  scanf("%g", &p);
  test=Fquan(1,DOFr,p);
  printf("EFFECT          SS        DOF        MS        F
    SIGNIF\n");
  printf("------          --        ---        --        -
    ------\n");
```

```c
fcr (i=0; i<(pow(2,k)-1);++i)
  {
    q=SS[i].sum/(SSr/DOFr);
    printf("%s                    %g          1         %g
%g",SS[i].name,SS[i].sum,
    SS[i].sum,q);
    if (test<=q)
  printf("       YES\n");
    else
  printf("       NO\n");
  }

}
}
```

LIST OF REFERENCES

1. COMSURFWARDEVGRU Instruction 3360.2, *SHAREM Program Statistical Testing Guidelines*, 1990.

2. Larson, H. J., *Introduction to Probability Theory and Statistical Inference*, John Wiley & Sons, Inc., 1982.

3. Box, G.E.P., Hunter, W.G. & Hunter, J.S., *Statistics for Experimenters*, John Wiley & Sons, Inc., 1978.

4. Agresti, A., *Catagorical Data Analysis*, John Wiley & Sons, Inc., 1990.

5. Bruning, J.L. and Kintz, B.L., *Computational Handbook of Statistics*, Scott, Foresman and Co., 1968.

6. *Biometrika, Volume XXXIV*, Cambridge University Press, 1947.

DISTRIBUTION LIST

1. Defense Technical Information Center   2
   Cameron Station
   Alexandria, VA 22304-6145

2. Library, Code 52   2
   Naval Postgraduate School
   Monterey, CA 93943-5002

3. Professor Harold J. Larson   2
   Naval Postgraduate School
   Monterey, CA 93943-5002

4. Professor Glenn Lindsay   2
   Naval Postgraduate School
   Monterey, CA 93943-5002

5. Commander   2
   Surface Warfare Development Group
   (ATTN: LCDR Summer)
   Naval Amphibious Base, Little Creek
   Code N3
   Norfolk, VA 23521-5160

6. LT. Bill Ridings   1
   2601 Three Bridge Rd.
   Powhatan, VA 23139